A Unified Formal Model for Service Oriented Architecture to Enforce Security Contracts

Diana Allam

A Unified Formal Model for Service Oriented Architecture to Enforce Security Contracts

Diana Allam

ASCOLA group; EMN-INRIA, LINA
Dept. Informatique. Ecole des Mines de Nantes
4 rue Alfred Kastler, 44307 NANTES Cedex 3, France
diana.allam@mines-nantes.fr

Abstract

In this paper we introduce a model as a foundation for heterogeneous services, therefore unifying web services technologies in SOA (Service Oriented Architecture), specifically, SOAP/WS* and RESTful models. This model abstracts away from service implementations, in order to verify and to enforce some important security properties.

Categories and Subject Descriptors D2.4 [*Software Engineering*]: Software/Program Verification—Correctness proofs, Formal methods, Model checking, Programming by contract, Validation; D3.1 [*Programming Languages*]: Formal Definitions and Theory—Semantics, Syntax; D.3.3 [*Programming Languages*]: Language Constructs and Features—Abstract data types, Concurrent programming structures, Constraints, Data types and structure, Input/output, Recursion

General Terms Design , Languages, Security, Standardization, Verification

Keywords Formal Methods, Security, Reference Monitors, Aspects, Service-Oriented Architecture

1. Motivation

Service-based applications can be built according to the two most competing technologies: WS* and RESTful. Both have similar characteristics but architectural decisions and targeted applications are different, [8]. Today, SOAP/WS* and RESTful models are often supported at the same time: *e.g.*, SAP AG's Business ByDesign product is based on the SOAP/WS* model, while its Gateway product supports the composition of RESTful services. In such a heterogeneous system, a unified model has an importance in controlling exchanged messages.

Actually, despite the differences between WS* and RESTful technologies, we can unify them under similar concepts: distributed agents are built by means of interacting distributed services that are composed in a black-box manner using well-defined service interfaces. These agents provide services while requiring other services that are consumed.

Such a global abstraction based on the notion of black-boxes for SOA technologies provides significant benefits for the management of distributed services, notably their security properties that can be specified through contracts at the interaction level. Despite the existing security standards, some very skilled attacks are still discovered. Most of these attacks are due to weaknesses in security policies. Hence, the verification of security contracts becomes a fundamental need in SOA environments. Security vulnerabilities appear at different levels. Here, we are only interested in security control at the message exchanges level.

2. Background and related work

A wide variety of formal models exists for service-oriented computing. Two distinguished approaches of formalization are presented: process calculus models for expressing and analyzing service based-systems, [10], or models for giving a formal semantic for a standard orchestration language, like BPEL, [7]. The drawback of these orchestration models is that they present a service implementation formalism for local processes description, which complicates the model with multiple communication rules. While verifying some important security properties, like access control on services, confidentiality on transmitted data and authentication for access rights, does not require such complex models. Instead, a simple model hiding the local implementation details is more adequate for security control mediating message exchanges. Such a model is called black-box model, like [9].

Regarding works on formal treatment of security properties for service-based systems and their enforcement, they occupy a big part of the state of the art, like in [3, 4, 6].

AOSD'12, Mar 25-30, 2012, Potsdam, Germany
ACM 978-1-4503-1092-5/12/03.

In these models, the security control is tightly linked to the concrete implementation and to the used language. Consequently, approaches of the former classes are often subject to the problem that they may not well support enforcement of security properties due to the gap of abstraction between the formal model and concrete implementations, [3]. Such constraint is inadequate with the language-independent (loosely coupling) property in SOA for security enforcement. Based on black-box models, security control could be provided by a wrapper monitor, like in [5].

3. Approach and uniqueness

According to a global contract, each partner in a communication deduces by projection a specification of the functionalities that it must locally implement. Of course, all these projections must ensure that the local functionalities, once gathered, effectively collaborate to realize the global contract. For each partner, it remains to realize the projection, by a local implementation. The ANR project CESSA [1] aims at tackling this synthesis problem, following a *top-down* approach, from descriptions to realizations. This project seeks an abstract framework for reasoning about security properties of SOA based web services, independently from implementation details. Thus, we distinguish two parts:

- First, we seek a formal model for service-oriented computing that enables to simplify the expression of fundamental security properties. In accordance with black-box models, we seek a message-passing model, like [9].
- Second, a security model must be built over the first one to apply security contracts at message exchanges level. For that, we seek a new notion of distributed reference monitors which assure to maintain the projected contract on the local encapsulated agent. Implementation of such monitors would be by using aspects language.

4. Results and contributions

Here we briefly describe our contribution on the first part of CESSA, the functional formal model. We have a simple formal model hiding the local details which makes it independent from implementation languages used for describing processes execution. With such a black-box model, the communication level over the network is abstracted with three main rules: (i) [LOC]: abstracts local executions that consume incoming messages and produce messages to be sent over the network; (ii) [OUT]: enables the passage of a message throughout the agent interface to the network; and (iii) [IN]: enables the receipt of a message from the network to the designated agent throughout its interface.

Figure 1 illustrates our formal model compared to the two technologies: WS* and RESTful. Our model is message-based with a chemical semantic, [2]. Web services are viewed as abstract agents exchanging messages via the network. Services are available thanks to the notion of communication channels. Messages can carry channels, thus ensuring full channel mobility. This model supports a sound type system avoiding type-based communication errors.

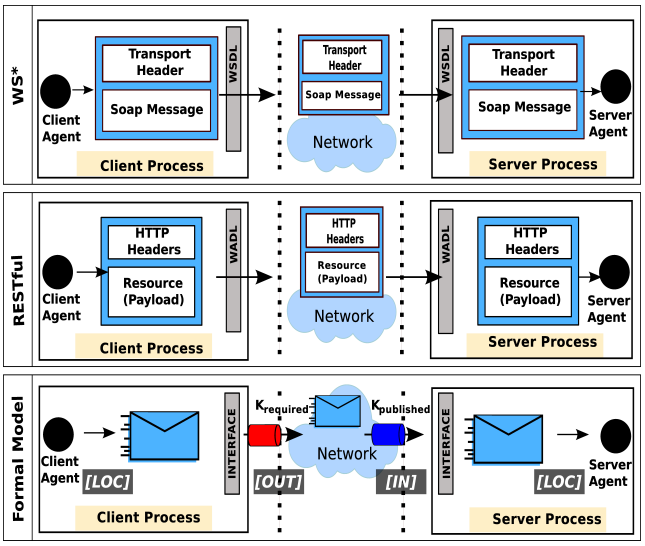

Figure 1. Relationship between WS*/RESTful and our formal model

Future work will demonstrate that this model is able to cope with service contracts, when they are based on the content of messages, especially security contracts. Security policies will be defined using our specific language, CSPL (CESSA Security Policy Language), [1].

References

[1] Cessa (compositional evolution of secure services using aspects). http://cessa.gforge.inria.fr.

[2] G. Berry and G. Boudol. The chemical abstract machine. *Theoretical Computer Science*, 96(1), 1992.

[3] K. Bhargavan and al. Modular verification of security protocol code by typing. In *POPL*, 2010.

[4] K. Bhargavan, C. Fournet, and A. Gordon. Verifying policy-based security for web services. In *CCS*, 2004.

[5] A. Chander, D. Dean, and J. Mitchell. A distributed high assurance reference monitor. In *ISC*, 2004.

[6] C. Fournet, A. Gordon, and S. Maffeis. A type discipline for authorization in distributed systems. In *CCS*, 2007.

[7] R. Lucchi and M. Mazzara. A pi-calculus based semantics for ws-bpel. In *Journal of Logic and Algebraic Programming*. Elsevier press, 2005.

[8] C. Pautasso, O. Zimmermann, and F. Leymann. Restful web services vs. big web services: making the right architectural decision. In *Proceedings of the 17th International Conference on World Wide Web*, pages 805–814. ACM, 2008.

[9] F. Seehusen and K. Stolen. Information flow security, abstraction and composition. *IETF Information and Security*, 3(1):9–33, 2009.

[10] H. T. Vieira, L. Caires, and J.o C. Seco. The conversation calculus: a model of service oriented computation. In *In Proc. of ESOP'08, LNCS*. Springer, 2008.